**Generare permutari -**  
#include <iostream>  
using namespace std;  
  
int x[100],n;  
  
***int are\_succ(int k)***  
{ int as;  
if (x[k]<n) /\* verific daca mai am elemente in multime pe nivelul k\*/  
{x[k]++; /\* x[k]=x[k]+relatia de ordine\*/  
as=1;}  
else as=0;  
return as;  
}  
  
***int e\_valid(int k)***  
{ int ok=1,i; /\*ok=1 presupun ca sunt indeplinite conditiile interne in cazul de fata presupun ca sunt distincte \*/  
for(i=1;i<=k-1;i++)  
if(x[i]==x[k])ok=0;  
return ok;  
}  
  
***void afis()***  
{ int i;  
for(i=1;i<=n;i++)  
cout<<x[i]<<" ";  
cout<<endl;  
}  
  
***void back(int k)***  
  
{ int as,ev;  
x[k]=0; /\* initializez cu primul element care nu apartine multimii conform relatiei de ordine\*/  
/\*stiva nu este vida nu s-au terminat elementele multimii A1  
caut un succesor valid pentru elementul de pe pozitia k\*/  
***while(k>0)***  
***{ do***  
{  
as=are\_succ(k);  
if(as) ev=e\_valid(k);}  
***while(as && !ev);* caut cat timp are succesor si nu e valid sau nu mai are succesor**  
if(as )  
{ **if( k==n) afis();**  
else  
{ k++;urc in stiva- **pas inainte pentru completarea vectorului solutie**  
x[k]=0;  
/\* anulez valorile care au fost eventual generate anterior cf. rel. de ordine a.i. adunand relatia de ordine sa ajung la primul element din multime\*/  
}  
}  
else k--; /\* cobor in stiva **pas inapoi- revenire pentru cautarea unui alt element** //\*/  
/\* sf. while generare solutii posibile\*/  
}  
}  
  
**int main()**  
{  
cin>>n;  
back(1);  
return 0;  
  
}